

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG  
KHOA AN TOÀN THÔNG TIN**



**BÁO CÁO BÀI THỰC HÀNH  
HỌC PHẦN: THỰC TẬP CƠ SỞ  
MÃ HỌC PHẦN: INT13147**

**BÀI THỰC HÀNH 4.1  
LẬP TRÌNH CLIENT/SERVER ĐỂ TRAO ĐỔI  
THÔNG TIN AN TOÀN**

Sinh viên thực hiện:

B22DCAT253 – Đinh Thị Thanh Tâm

Giảng viên hướng dẫn: TS. Đinh Trường Duy

**HỌC KỲ 2 NĂM HỌC 2024-2025**

# MỤC LỤC

MỤC LỤC.....	2
DANH MỤC CÁC HÌNH VẼ.....	3
DANH MỤC CÁC TỪ VIẾT TẮT.....	4
<b>CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH</b> .....	<b>5</b>
1.1 Mục đích.....	5
1.2 Tìm hiểu lý thuyết .....	5
1.2.1 Socket .....	5
1.2.2 Giao thức TCP.....	6
1.2.3 Các thành phần trong lập trình socket TCP.....	7
1.2.4 Quy trình thiết lập kết nối TCP (3-way handshake).....	7
1.2.5 Các hàm và thao tác cơ bản.....	7
1.2.6 Port và IP .....	8
<b>CHƯƠNG 2. NỘI DUNG THỰC HÀNH</b> .....	<b>9</b>
2.1 Chuẩn bị môi trường .....	9
2.2 Các bước thực hiện.....	9
2.2.1 Lập trình client và server với TCP socket.....	9
2.2.2 Trao đổi thông điệp giữa client và server và đảm bảo tính toàn vẹn của thông điệp khi trao đổi.....	11
KẾT LUẬN .....	16
TÀI LIỆU THAM KHẢO .....	16

## DANH MỤC CÁC HÌNH VẼ

Hình 1 Khởi chạy server .....	10
Hình 2 Khởi chạy Client .....	10
Hình 3 Thông điệp Server nhận được gửi từ Client .....	10
Hình 4 Thông điệp Client nhận được từ Server .....	10
Hình 5 Kết quả bắt gói tin trao đổi giữa Server và Client bằng Wireshark .....	11
Hình 6 Sửa đổi file server.py.....	11
Hình 7 Sửa đổi file client.py .....	12
Hình 8 Chạy server.....	12
Hình 9 Sau đó chạy client. ....	13
Hình 10 Thông điệp Server nhận được từ Client .....	13
Hình 11 Thông điệp Client nhận được từ Server .....	13
Hình 12 Kết quả bắt gói tin trao đổi giữa Server và Client bằng Wireshark .....	15

## DANH MỤC CÁC TỪ VIẾT TẮT

<b>Từ viết tắt</b>	<b>Thuật ngữ tiếng Anh/Giải thích</b>	<b>Thuật ngữ tiếng Việt/Giải thích</b>
IP	Internet Protocol	Giao thức Internet
UDP	User Datagram Protocol	Giao thức Datagram người dùng
HTTP	HyperText Transfer Protocol	Giao thức truyền tải siêu văn bản
HTTPS	HTTP Secure / HTTP over TLS	Giao thức HTTP bảo mật
FTP	File Transfer Protocol	Giao thức truyền tập tin
SSH	Secure Shell	Giao thức điều khiển từ xa an toàn

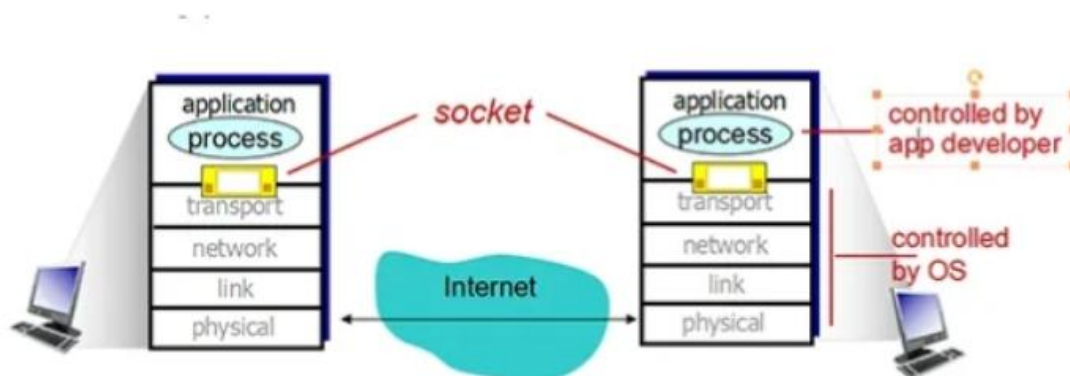
# CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH

## 1.1 Mục đích

Sinh viên hiểu về cơ chế client/server và có thể tự lập trình client/server dựa trên socket, sau đó thực hiện ca đặt giao thức đơn giản để trao đổi thông tin an toàn.

## 1.2 Tìm hiểu lý thuyết

### 1.2.1 Socket



#### 1.2.1.1 Socket là gì?

Socket là điểm cuối end-point trong liên kết truyền thông hai chiều (two-way communication) biểu diễn kết nối giữa Client – Server. Các lớp Socket được ràng buộc với một cổng port (thể hiện là một con số cụ thể) để các tầng TCP (TCP Layer) có thể định danh ứng dụng mà dữ liệu sẽ được gửi tới.

Socket là giao diện lập trình ứng dụng mạng được dùng để truyền và nhận dữ liệu trên internet. Giữa hai chương trình chạy trên mạng cần có một liên kết giao tiếp hai chiều, hay còn gọi là two-way communication để kết nối 2 process trò chuyện với nhau. Điểm cuối (endpoint) của liên kết này được gọi là socket. Một chức năng khác của socket là giúp các tầng TCP hoặc TCP Layer định danh ứng dụng mà dữ liệu sẽ được gửi tới thông qua sự ràng buộc với một cổng port (thể hiện là một con số cụ thể), từ đó tiến hành kết nối giữa client và server.

#### 1.2.1.2 Tầm quan trọng

Người dùng cần đến socket vì nó là một cách tiêu biểu để thiết lập và quản lý kết nối mạng giữa các thiết bị và ứng dụng trên Internet. Dưới đây là một số lý do chính:

- **Giao tiếp mạng:** Socket cho phép ứng dụng gửi và nhận dữ liệu qua mạng, cho phép giao tiếp giữa các thiết bị từ xa. Điều này là cần thiết trong nhiều ứng dụng như trò chơi trực tuyến, chat, truyền tệp, và nhiều ứng dụng mạng khác.
- **Tính linh hoạt:** Socket hỗ trợ nhiều loại kết nối mạng khác nhau như TCP, UDP, và các giao thức khác, cung cấp sự linh hoạt trong cách ứng dụng giao tiếp với nhau.

- **Phân phối dữ liệu:** Sử dụng socket, dữ liệu có thể được phân phối từ một nguồn tới nhiều đích hoặc ngược lại. Điều này cho phép triển khai các ứng dụng mạng phức tạp như streaming video, trò chơi trực tuyến, và các ứng dụng đa người dùng.
- **Tương thích đa nền tảng:** Socket có sẵn trên nhiều nền tảng và hệ điều hành khác nhau, từ máy tính cá nhân đến thiết bị di động, giúp cho việc phát triển ứng dụng mạng trở nên dễ dàng và linh hoạt hơn.
- **Kiểm soát độ trễ:** Sử dụng socket, người dùng có thể kiểm soát và tối ưu hóa độ trễ trong việc truyền và nhận dữ liệu qua mạng, điều này quan trọng đặc biệt trong các ứng dụng đòi hỏi độ trễ thấp như trò chơi trực tuyến và ứng dụng thời gian thực.

Tóm lại, socket là một công cụ quan trọng trong việc phát triển các ứng dụng mạng, cho phép giao tiếp hiệu quả và đa dạng giữa các thiết bị và ứng dụng trên Internet.

#### *1.2.1.3 Phân loại Socket*

Socket có thể được phân loại chủ yếu dựa trên hai tiêu chí: giao thức và cách sử dụng. Dưới đây là một số phân loại phổ biến của socket:

*Theo giao thức:*

- **Socket TCP (Transmission Control Protocol):** Sử dụng giao thức TCP để thiết lập kết nối đáng tin cậy, có kiểm soát lỗi và bảo đảm dữ liệu đến được đích một cách chính xác.
- **Socket UDP (User Datagram Protocol):** Sử dụng giao thức UDP, không đảm bảo tính toàn vẹn hoặc độ tin cậy, nhưng thích hợp cho các ứng dụng cần truyền dữ liệu nhanh mà không cần quá nhiều kiểm soát.

*Theo cách sử dụng:*

- **Socket Server:** Là socket được sử dụng để lắng nghe và chấp nhận kết nối từ các client khác.
- **Socket Client:** Là socket được sử dụng để thiết lập kết nối và truyền dữ liệu đến một socket server.

Mỗi loại socket có mục đích và ứng dụng khác nhau trong lập trình mạng. Sử dụng đúng loại socket sẽ giúp đảm bảo tính ổn định và hiệu suất của ứng dụng mạng.

#### *1.2.2 Giao thức TCP*

TCP là một giao thức ở tầng vận chuyển (Transport Layer) trong mô hình TCP/IP. Giao thức này cung cấp một kênh truyền dữ liệu đáng tin cậy, có kiểm tra lỗi, đảm bảo thứ tự và không mất mát gói tin. TCP được gọi là giao thức hướng kết nối (connection-oriented), có nghĩa là trước khi gửi dữ liệu, hai bên (client và server) phải thiết lập một kết nối ổn định.

Các đặc điểm nổi bật của TCP bao gồm:

- Truyền dữ liệu theo luồng (stream).
- Đảm bảo thứ tự các gói tin.
- Đảm bảo không mất dữ liệu.

- Có cơ chế kiểm soát lỗi, kiểm soát lưu lượng và xử lý tắc nghẽn.

### **1.2.3 Các thành phần trong lập trình socket TCP**

#### **1.2.3.1 Server (máy chủ)**

Server là phía "nghe" các kết nối đến từ client. Nó phải được thiết lập để:

- Khởi tạo socket.
- Gán socket với một địa chỉ IP và port cụ thể.
- Lắng nghe các yêu cầu kết nối.
- Chấp nhận kết nối từ client.
- Thực hiện trao đổi dữ liệu.
- Đóng kết nối khi hoàn tất.

#### **1.2.3.2 Client (máy khách)**

Client là phía khởi tạo kết nối đến server. Quá trình hoạt động của client thường gồm:

- Tạo socket.
- Kết nối đến địa chỉ IP và cổng của server.
- Gửi và nhận dữ liệu.
- Đóng kết nối.

### **1.2.4 Quy trình thiết lập kết nối TCP (3-way handshake)**

Khi socket của tiến trình khách vừa được tạo, TCP trên máy khách sẽ tiến hành thực hiện quá trình bắt tay 3 bước và thiết lập kết nối TCP tới máy chủ.

Trong quá trình bắt tay 3 bước, khi tiến trình chủ nhận thấy tiến trình khách, nó sẽ tự tạo ra 1 socket mới chỉ dành riêng cho tiến trình khách đó. Khi được máy khách gõ cửa, chương trình kích hoạt với phương thức accept(). Cuối quá trình bắt tay 3 bước, một kết nối TCP tồn tại giữa socket của máy khách và socket của máy chủ.

### **1.2.5 Các hàm và thao tác cơ bản**

<b>Hàm (function)</b>	<b>Chức năng</b>
socket()	Tạo một socket mới.
bind()	Gán địa chỉ IP và port cho socket (dùng cho server).
listen()	Đưa socket vào trạng thái chờ kết nối (server).
accept()	Chấp nhận một kết nối từ client.
connect()	Yêu cầu kết nối đến server (client).
send()	Gửi dữ liệu.
recv()	Nhận dữ liệu.
close()	Đóng kết nối socket.

### ***1.2.6 Port và IP***

- IP (Internet Protocol address): Địa chỉ của máy tính trong mạng.
- Port: Số hiệu giúp định danh ứng dụng mạng đang chạy trên máy đó. Ví dụ:
  - HTTP: 80
  - HTTPS: 443
  - FTP: 21
  - SSH: 22

Tổng cộng có 65.535 port, được chia làm:

- Well-known ports: 0 - 1023 (dành cho các dịch vụ chuẩn).
- Registered ports: 1024 - 49151.
- Dynamic/Private ports: 49152 - 65535.



## CHƯƠNG 2. NỘI DUNG THỰC HÀNH

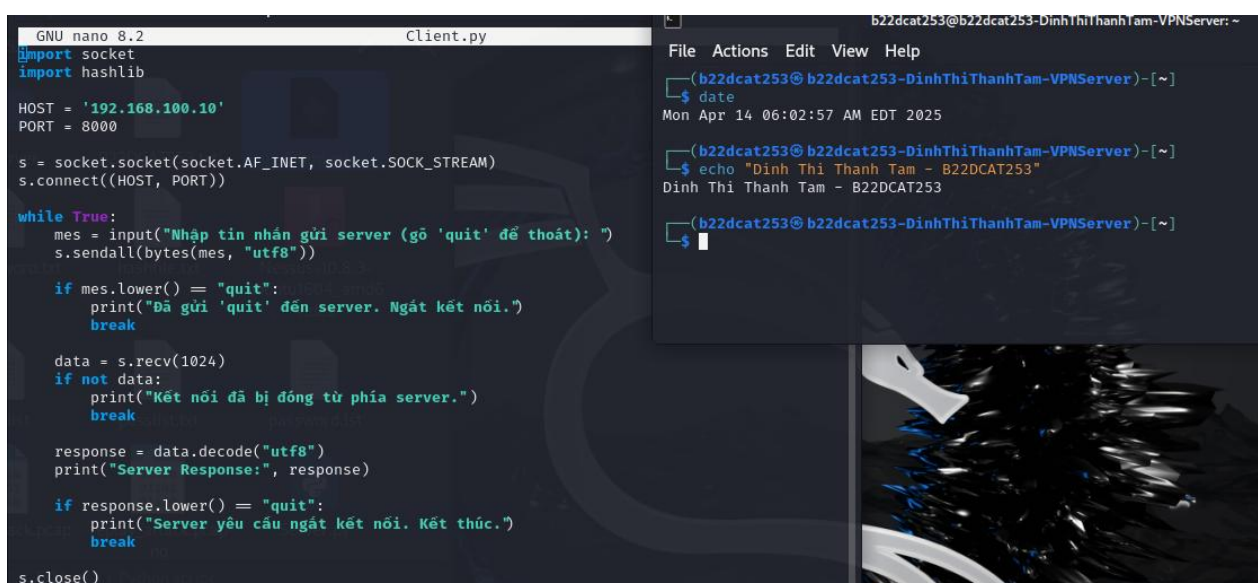
### 2.1 Chuẩn bị môi trường

- Môi trường Python để chạy được ứng dụng client/server đã lập trình.
- Phần mềm Wireshark.
- Máy ảo Kali Linux để chạy server có địa chỉ ip: 192.168.100.10
- Máy ảo Kali Linux để chạy client.

### 2.2 Các bước thực hiện

#### 2.2.1 Lập trình client và server với TCP socket

- Lập trình client.



```
GNU nano 8.2 Client.py
import socket
import hashlib

HOST = '192.168.100.10'
PORT = 8000

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))

while True:
    mes = input("Nhập tin nhắn gửi server (gõ 'quit' để thoát): ")
    s.sendall(bytes(mes, "utf8"))

    if mes.lower() == "quit":
        print("Đã gửi 'quit' đến server. Ngắt kết nối.")
        break

    data = s.recv(1024)
    if not data:
        print("Kết nối đã bị đóng từ phía server.")
        break

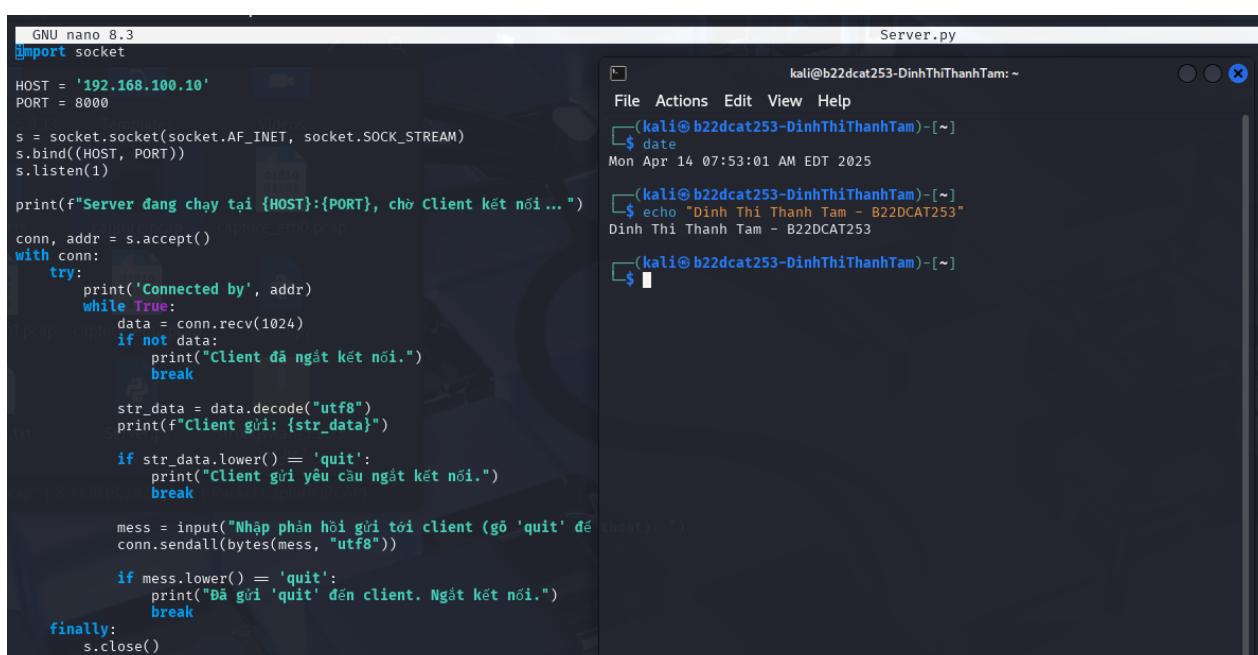
    response = data.decode("utf8")
    print("Server Response:", response)

    if response.lower() == "quit":
        print("Server yêu cầu ngắt kết nối. Kết thúc.")
        break

s.close()
```

```
b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~
File Actions Edit View Help
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ date
Mon Apr 14 06:02:57 AM EDT 2025
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$
```

- Lập trình server.



```
GNU nano 8.3 Server.py
import socket

HOST = '192.168.100.10'
PORT = 8000

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)

print(f"Server đang chạy tại {HOST}:{PORT}, chờ Client kết nối...")

conn, addr = s.accept()
with conn:
    try:
        print('Connected by', addr)
        while True:
            data = conn.recv(1024)
            if not data:
                print("Client đã ngắt kết nối.")
                break

            str_data = data.decode("utf8")
            print(f"Client gửi: {str_data}")

            if str_data.lower() == 'quit':
                print("Client gửi yêu cầu ngắt kết nối.")
                break

            mess = input("Nhập phản hồi gửi tới client (gõ 'quit' để ngắt kết nối): ")
            conn.sendall(bytes(mess, "utf8"))

            if mess.lower() == 'quit':
                print("Đã gửi 'quit' đến client. Ngắt kết nối.")
                break

    finally:
        s.close()
```

```
kali@b22dcat253-DinhThiThanhTam: ~
File Actions Edit View Help
(kali@b22dcat253-DinhThiThanhTam)-[~]
$ date
Mon Apr 14 07:53:01 AM EDT 2025
(kali@b22dcat253-DinhThiThanhTam)-[~]
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
(kali@b22dcat253-DinhThiThanhTam)-[~]
$
```

- Kết nối server với client và thực hiện gửi thông điệp.

The image shows two terminal windows. The left window, titled 'kali@b22dcat253-DinhThiThanhTam: ~', shows the execution of 'python3 Server.py'. The output indicates the server is running at 192.168.100.10:8000 and waiting for a client. The right window, titled 'kali@b22dcat253-DinhThiThanhTam: ~', shows the execution of 'date' (displaying 'Mon Apr 14 07:53:01 AM EDT 2025') and 'echo "Dinh Thi Thanh Tam - B22DCAT253"' (displaying 'Dinh Thi Thanh Tam - B22DCAT253').

Hình 1 Khởi chạy server

The image shows two terminal windows. The left window, titled 'b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~', shows the execution of 'python3 Client.py'. The prompt 'Nhập tin nhắn gửi server (gõ 'quit' để thoát):' is shown. The right window, titled 'b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~', shows the execution of 'date' (displaying 'Mon Apr 14 06:02:57 AM EDT 2025') and 'echo "Dinh Thi Thanh Tam - B22DCAT253"' (displaying 'Dinh Thi Thanh Tam - B22DCAT253').

Hình 2 Khởi chạy Client

- Client gửi thông điệp cá nhân hóa cho server: “Hello, I am B22DCAT253 client”.

The image shows two terminal windows. The left window, titled 'b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~', shows the execution of 'python3 Client.py'. The prompt 'Nhập tin nhắn gửi server (gõ 'quit' để thoát):' is shown, and the user has entered 'Hello, I am B22DCAT253 client'. The right window, titled 'b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~', shows the execution of 'date' (displaying 'Mon Apr 14 06:02:57 AM EDT 2025') and 'echo "Dinh Thi Thanh Tam - B22DCAT253"' (displaying 'Dinh Thi Thanh Tam - B22DCAT253').

- Server nhận được hiển thị thông điệp nhận được và gửi lại client thông điệp: server gửi lại “Hello, I am B22DCAT253 server”.

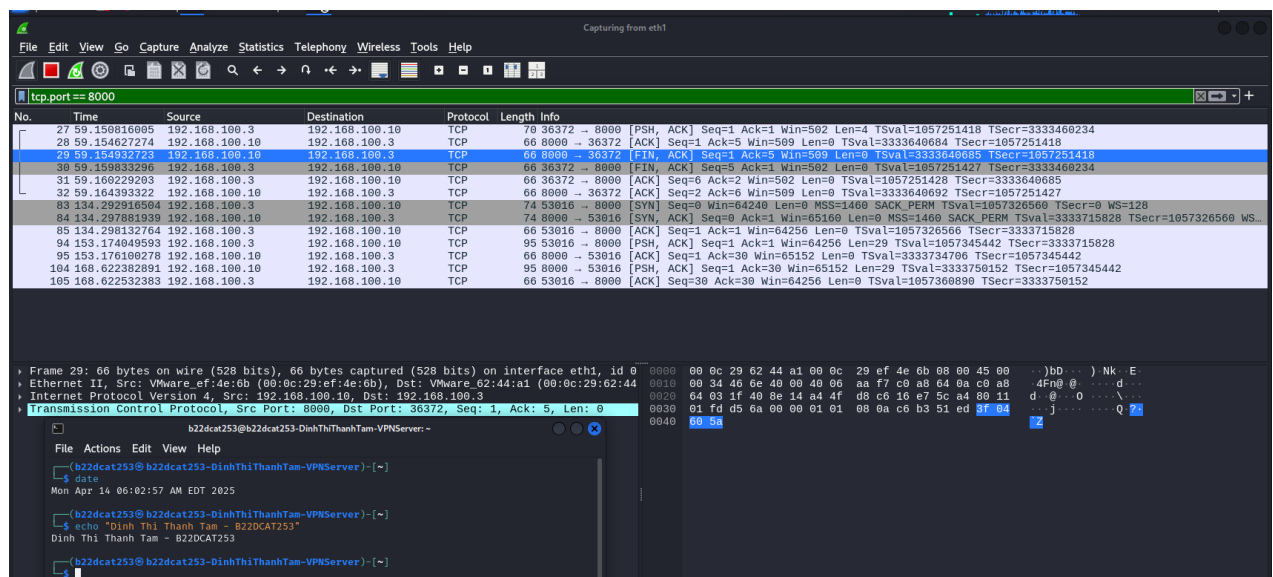
The image shows two terminal windows. The left window, titled 'kali@b22dcat253-DinhThiThanhTam: ~', shows the execution of 'python3 Server.py'. The output indicates the server is running at 192.168.100.10:8000 and waiting for a client. The right window, titled 'kali@b22dcat253-DinhThiThanhTam: ~', shows the execution of 'date' (displaying 'Mon Apr 14 07:53:01 AM EDT 2025') and 'echo "Dinh Thi Thanh Tam - B22DCAT253"' (displaying 'Dinh Thi Thanh Tam - B22DCAT253').

Hình 3 Thông điệp Server nhận được gửi từ Client

The image shows two terminal windows. The left window, titled 'b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~', shows the execution of 'python3 Client.py'. The prompt 'Nhập tin nhắn gửi server (gõ 'quit' để thoát):' is shown, and the user has entered 'Hello, I am B22DCAT253 client'. The right window, titled 'b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~', shows the execution of 'date' (displaying 'Mon Apr 14 06:02:57 AM EDT 2025') and 'echo "Dinh Thi Thanh Tam - B22DCAT253"' (displaying 'Dinh Thi Thanh Tam - B22DCAT253').

Hình 4 Thông điệp Client nhận được từ Server

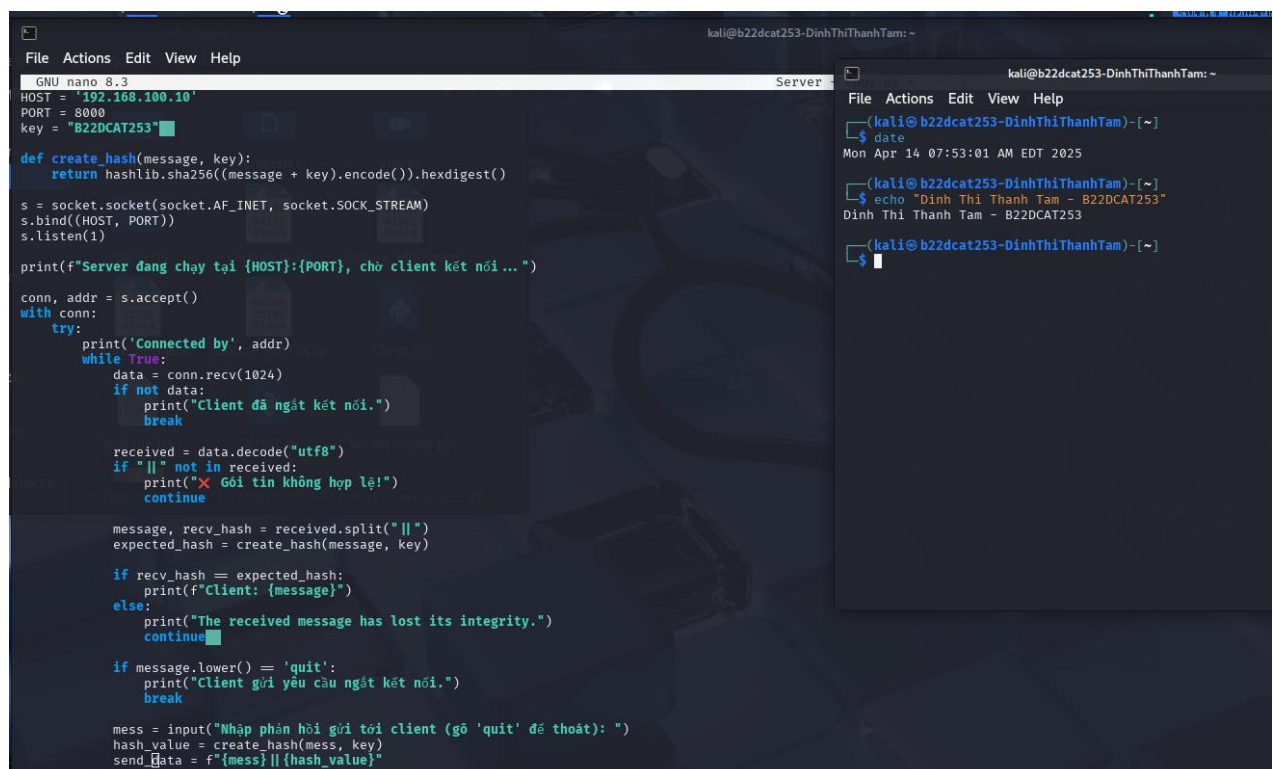
- Sử dụng Wireshark để bắt các thông tin đã gửi từ client đến server và ngược lại, lọc gói tin: `tcp.port == 8000`



Hình 5 Kết quả bắt gói tin trao đổi giữa Server và Client bằng Wireshark

### 2.2.2 Trao đổi thông điệp giữa client và server và đảm bảo tính toàn vẹn của thông điệp khi trao đổi

- Từ client và server, sửa đổi để sao cho: khi gửi thông điệp sẽ gửi kèm theo giá trị băm của (thông điệp+key) để phía bên kia kiểm tra xác minh tính toàn vẹn.
- Thông nhất key là B22DCAT253.
- Sửa đổi file `py`



Hình 6 Sửa đổi file `server.py`





```
File Actions Edit View Help
GNU nano 8.2
import socket
import hashlib

HOST = '192.168.100.10'
PORT = 8000
key = "B22DCAT253"

def create_hash(message, key):
    return hashlib.sha256((message + key).encode()).hexdigest()

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))

while True:
    mes = input("Nhập tin nhắn gửi server (gõ 'quit' để thoát): ")
    hash_value = create_hash(mes, key)
    send_data = f"{mes}||{hash_value}"
    s.sendall(send_data.encode("utf8"))

    if mes.lower() == "quit":
        print("Đã gửi 'quit' đến server. Ngắt kết nối.")
        break

    data = s.recv(1024)
    if not data:
        print("Kết nối đã bị đóng từ phía server.")
        break

    response_raw = data.decode("utf8")
    if "||" in response_raw:
        response, recv_hash = response_raw.split("||")
        if recv_hash == create_hash(response, key):
            print("Server Response:", response)
        else:
            print("The received message has lost its integrity.")
            continue
    else:
        print("⚠ Dữ liệu không hợp lệ từ server:", response_raw)

    if response.lower() == "quit":
        print("Server yêu cầu ngắt kết nối. Kết thúc.")
        break

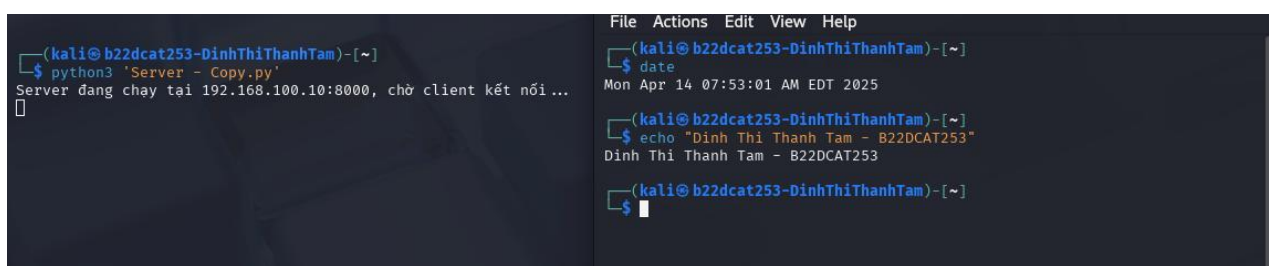
^G Help      ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line
```

```
File Actions Edit View Help
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServ)~
$ date
Mon Apr 14 06:02:57 AM EDT 2025

(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServ)~
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
```

Hình 7 Sửa đổi file client.py

- Kết nối server với client và thực hiện gửi thông điệp.



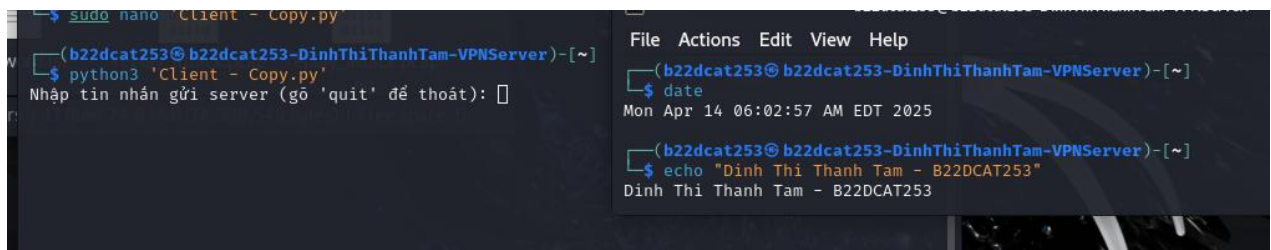
```
File Actions Edit View Help
(kali@b22dcat253-DinhThiThanhTam)~
$ python3 'Server - Copy.py'
Server đang chạy tại 192.168.100.10:8000, chờ client kết nối ...
^C
```

```
File Actions Edit View Help
(kali@b22dcat253-DinhThiThanhTam)~
$ date
Mon Apr 14 07:53:01 AM EDT 2025

(kali@b22dcat253-DinhThiThanhTam)~
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253

(kali@b22dcat253-DinhThiThanhTam)~
$
```

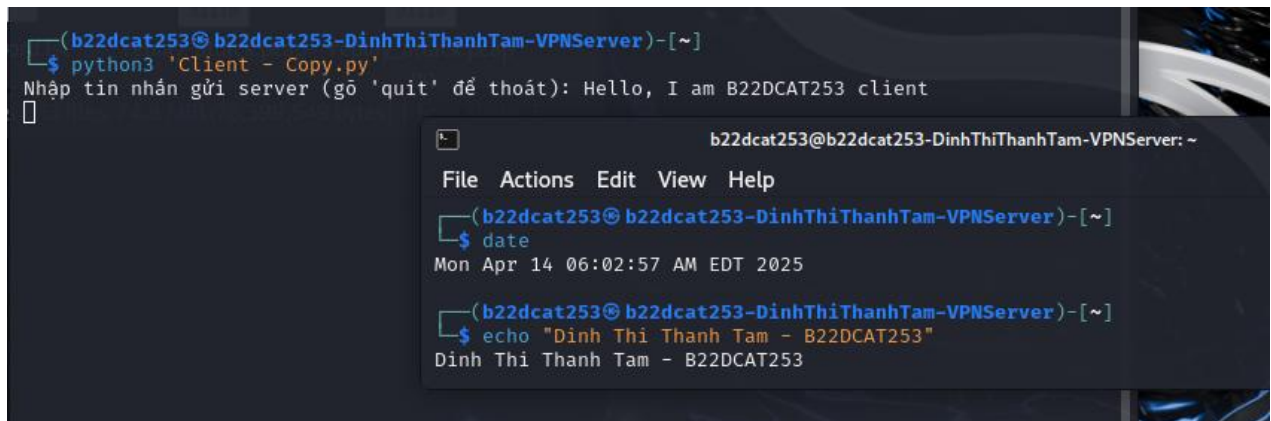
Hình 8 Chạy server.



```
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ python3 'Client - Copy.py'
Nhập tin nhắn gửi server (gõ 'quit' để thoát): 
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ date
Mon Apr 14 06:02:57 AM EDT 2025
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
```

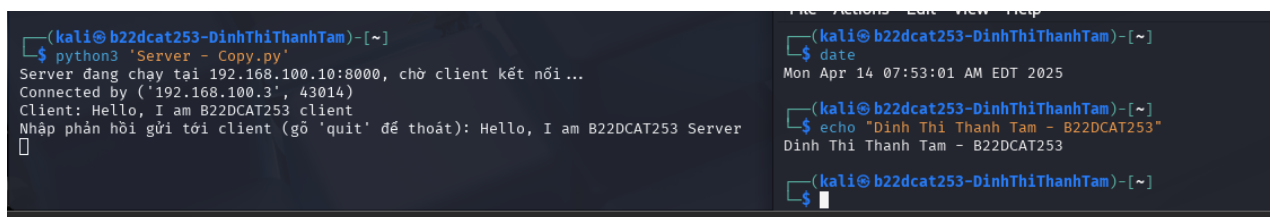
Hình 9 Sau đó chạy client.

- Client gửi thông điệp cá nhân hóa cho server: “Hello, I am B22DCAT253 client”.



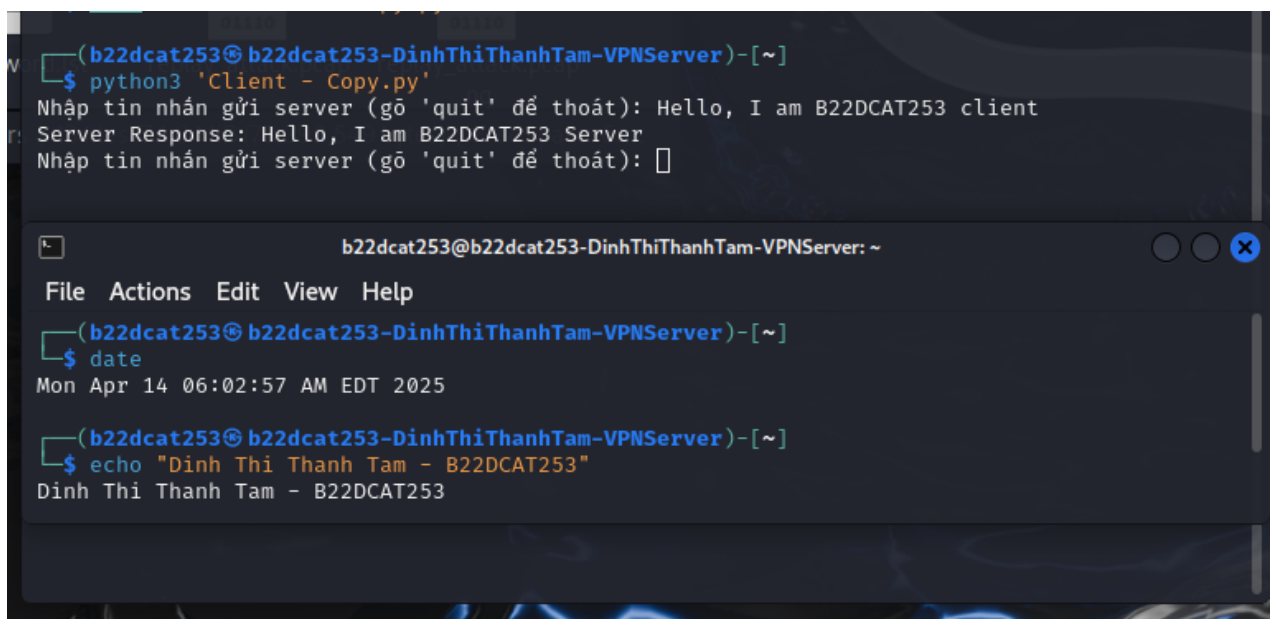
```
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ python3 'Client - Copy.py'
Nhập tin nhắn gửi server (gõ 'quit' để thoát): Hello, I am B22DCAT253 client
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ date
Mon Apr 14 06:02:57 AM EDT 2025
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
```

- Server nhận được hiển thị thông điệp nhận được và gửi lại client thông điệp: server gửi lại “Hello, I am B22DCAT253 server”.



```
(kali@b22dcat253-DinhThiThanhTam)-[~]
$ python3 'Server - Copy.py'
Server đang chạy tại 192.168.100.10:8000, chờ client kết nối...
Connected by ('192.168.100.3', 43014)
Client: Hello, I am B22DCAT253 client
Nhập phản hồi gửi tới client (gõ 'quit' để thoát): Hello, I am B22DCAT253 Server
(kali@b22dcat253-DinhThiThanhTam)-[~]
$ date
Mon Apr 14 07:53:01 AM EDT 2025
(kali@b22dcat253-DinhThiThanhTam)-[~]
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
(kali@b22dcat253-DinhThiThanhTam)-[~]
$
```

Hình 10 Thông điệp Server nhận được từ Client



```
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ python3 'Client - Copy.py'
Nhập tin nhắn gửi server (gõ 'quit' để thoát): Hello, I am B22DCAT253 client
Server Response: Hello, I am B22DCAT253 Server
Nhập tin nhắn gửi server (gõ 'quit' để thoát): 
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ date
Mon Apr 14 06:02:57 AM EDT 2025
(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
```

Hình 11 Thông điệp Client nhận được từ Server

- Thay đổi giá trị key của client thành B22DCAT253\_wrong và thực hiện gửi lại.

```

b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer: ~
File Actions Edit View Help
GNU nano 8.2 Client
import socket
import hashlib

HOST = '192.168.100.10'
PORT = 8000
key = "B22DCAT253_wrong"

def create_hash(message, key):
    return hashlib.sha256((message + key).encode()).hexdigest()

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))

while True:
    mes = input("Nhập tin nhắn gửi server (gõ 'quit' để thoát): ")
    hash_value = create_hash(mes, key)
    send_data = f"{mes}||{hash_value}"
    s.sendall(send_data.encode("utf8"))

    if mes.lower() == "quit":
        print("Đã gửi 'quit' đến server. Ngắt kết nối.")
        break

    data = s.recv(1024)
    if not data:
        print("Kết nối đã bị đóng từ phía server.")
        break

    response_raw = data.decode("utf8")
    if "||" in response_raw:
        response, recv_hash = response_raw.split("||")
        if recv_hash == create_hash(response, key):
            print("Server Response:", response)
        else:
            print("The received message has lost its integrity.")
            continue
    else:
        print("⚠ Dữ liệu không hợp lệ từ server:", response_raw)

    if response.lower() == "quit":
        print("Server yêu cầu ngắt kết nối. Kết thúc.")
        break

^G Help      ^O Write Out  ^F Where Is   ^K Cut        ^T Execute   ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify   ^/_ Go To Line
  
```

- Khởi chạy Client, gửi thông điệp đến Server

```

(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ sudo nano 'Client - Copy.py'

(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
$ python3 'Client - Copy.py'
Nhập tin nhắn gửi server (gõ 'quit' để thoát): Hello, I am B22DCAT253 Client

(b22dcat253@b22dcat253-DinhThiThanhTam-VPNServer)-[~]
File Actions Edit View Help
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
$
  
```

- Không đáp ứng tính toàn vẹn, server thông báo: “The received message has lost its integrity.”

```

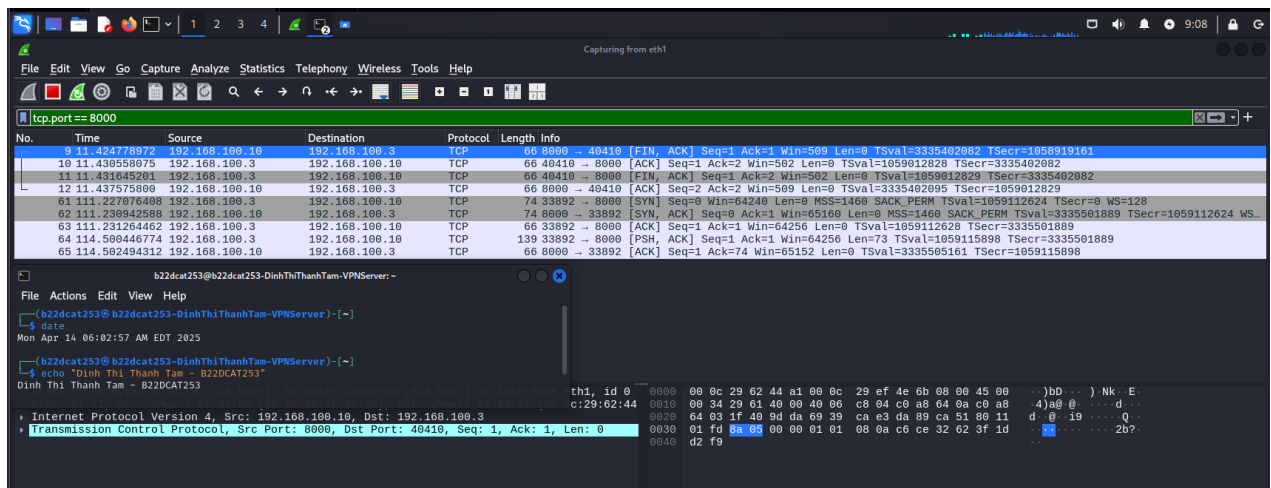
(kali@b22dcat253-DinhThiThanhTam)-[~]
$ python3 'Server - Copy.py'
Server đang chạy tại 192.168.100.10:8000, chờ client kết nối...
Connected by ('192.168.100.3', 40410)
The received message has lost its integrity.

(kali@b22dcat253-DinhThiThanhTam)-[~]
$ date
Mon Apr 14 07:53:01 AM EDT 2025

(kali@b22dcat253-DinhThiThanhTam)-[~]
$ echo "Dinh Thi Thanh Tam - B22DCAT253"
Dinh Thi Thanh Tam - B22DCAT253
  
```

- Bắt được các bản tin trao đổi giữa client và server trong Wireshark.





Hình 12 Kết quả bắt gói tin trao đổi giữa Server và Client bằng Wireshark

## KẾT LUẬN

- Lập trình và chạy thành công TCP Socket Client/Server
- Trao đổi thông điệp giữa client và server và đảm bảo tính toàn vẹn của thông điệp khi trao đổi
- Bắt được các bản tin trao đổi giữa client và server trong Wireshark

## TÀI LIỆU THAM KHẢO

[1] Tài liệu: Chapter 2: Application Layer V8.1 (9/2020)

[http://gaia.cs.umass.edu/kurose\\_ross/ppt.php](http://gaia.cs.umass.edu/kurose_ross/ppt.php)

[2] Tham khảo tài liệu: Chapter 2: Application Layer V8.1 (9/2020) tại địa chỉ [http://gaia.cs.umass.edu/kurose\\_ross/ppt.php](http://gaia.cs.umass.edu/kurose_ross/ppt.php) (chú ý ví dụ từ trang 105).